AvecVoting: Anonymous and Verifiable E-voting with Untrustworthy Counters on Blockchain

Meiqi Li*, Xinyi Luo*, Wentuo Sun*, Jian Li*†, Kaiping Xue*†‡
* School of Cyber Science and Technology, University of Science and Technology of China, Hefei, Anhui 230027, China
† School of Information Science and Technology, University of Science and Technology of China, Hefei, Anhui 230027, China
‡Corresponding author, kpxue@ustc.edu.cn

ICC 2022 - IEEE International Conference on Communications | 978-1-5386-8347-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICC45855.2022.9838840

***Abstract*—E-voting plays a vital role in modern social life. However, traditional e-voting systems usually rely on a trusted third party and therefore non-verifiable and prone to a single point of failure. In recent years, many researchers have tried to turn to blockchain to eliminate the vulnerabilities of e-voting systems. However, blockchain-based e-voting brings new problems in protecting voters' privacy and ballots' confidentiality, and causes a great performance degradation. In this paper, we propose AvecVoting, an anonymous and verifiable blockchain-based e-voting scheme, providing both strong security and high performance. Specifically, we utilize threshold encryption and one-time ring signature to protect voters' privacy and ballots' confidentiality. Furthermore, to improve the performance, we introduce the concept "counter" to count the ballots. Through the carefully designed** RandomSortition **and reputation-based** PayOff **algorithms based on smart contracts, AvecVoting can achieve correct counting even when some counters are untrustworthy. Our security and performance analyses show that AvecVoting provides strong security such as anonymity, non-repeatability, confidentiality, verifiability, etc., and meanwhile overcome the performance issues caused by blockchain and provides good efficiency in both voting and counting stages.**

***Index Terms*—E-voting, Blockchain, Smart Contract, One-time Ring Signature, Reputation.**

I. Introduction

Voting provides a justice and effective way of making decisions and thus plays a vital role in modern society. Traditional paper ballots require a lot of manpower and material resources, and both the voting and counting processes are cumbersome and redundant. On the contrary, electronic voting [1], or e-voting, is more convenient and efficient, and is not restricted by geographic distance. In modern life, e-voting is inseparable from people's lives. From national elections to as small as a group deciding who will be the leader or where to go to dinner, e-voting provides a very convenient way to make decisions.

Traditional e-voting protocols can be roughly divided into three categories: mix-net-based e-voting [2], homomorphic encryption-based e-voting [3], and blind signatures-based e-voting [4]. However, the above three traditional electronic voting protocols all require a trusted third party to collect, verify, and count ballots, which makes the system prone to a single point of failure. In 1994, Szabo [5] first proposed the concept of smart contracts which combine contracts with computer operations to allow traceable and irreversible transactions without the involvement of a third party. In 2008, the blockchain technology was officially born with Bitcoin [6], and prompted researchers to implement secure and reliable smart contracts based on blockchain, such as Ethereum and Hyperledger. Consequently, researchers began to utilize blockchain and smart contract technology to solve the problems existing in traditional e-voting, and established some decentralized e-voting systems without a trusted third party. However, the use of blockchain brings new problems in protecting voters' privacy and ballots' confidentiality, and causes great performance degradation [7].

Considering the above issues, our goal is to provide a blockchain-based e-voting system that meets both security and performance requirements. For security, the system should guarantee the security requirements in traditional e-voting, i.e., anonymity, non-repeatability, confidentiality, anti-tampering, verifiability, correctness, robustness. For performance, our system should complete voting and counting within acceptable time and computation overhead. To this end, we propose AvecVoting by adopting threshold encryption [8], [9] and one-time ring signature [10] algorithms for privacy and confidentiality. We further design a counter model for efficient and reliable counting. The main idea of the counter model is to set a group of "counters" to verify and count ballots off-chain, rather than directly counting in smart contracts with high computation overhead or requiring voters with low computing power to count themselves. By our carefully designed RandomSortition and reputation-based PayOff algorithms, AvecVoting can count the ballots correctly even when some counters are untrustworthy. In summary, this paper makes the following contributions:

1) We propose AvecVoting, a blockchain-based e-voting system that provides both strong security and high performance. We adopt threshold encryption and one-time ring signature to satisfy basic security requirements of e-voting systems, e.g., anonymity, confidentiality, non-repeatability.
2) To eliminate the performance degradation caused by the adoption of blockchain, we design a counter model for efficient and reliable counting. Further, to guarantee the security with untrustworthy counters, we propose RandomSortition and reputation-based PayOff algorithms, ensuring that honest counters get their due

Authorized licensed use limited to: University of Science & Technology of China. Downloaded on August 27,2022 at 06:55:52 UTC from IEEE Xplore. Restrictions apply.

reward while malicious counters are disqualified for counting and will be punished.

3) We give thorough security analysis to ensure AvecVoting can achieve anonymity, correctness, verifiability, and robustness. Besides, We evaluate the performance both the on-chain and off-chain parts of AvecVoting, and experiment results show that it can provide practically acceptable performance.

The rest of our paper is organized as follows. Section II introduces a series of work related to our scheme, and then in Section III, we introduce our system model and the security assumptions and the design goals. Then the proposed voting system is presented in detail in Section IV. Section V provides a security analysis and a performance analysis of our scheme. Finally, we have a summary of our work in Section VI.

II. Related Work

Existing e-voting protocols can be roughly divided into three categories: mix-net-based e-voting, homomorphic encryption-based e-voting, and blind signatures-based e-voting. Among them, mix-net-based e-voting [2] achieves anonymity by disrupting the correspondence between voters and ballots, and therefore a sufficient number of mix nodes are needed. Homomorphic encryption-based e-voting [3] realizes to directly perform operations on the ciphertext and obtain the final result without decrypting a single ballot, ensuring privacy. However, its computational complexity is relatively high. Blind signatures-based e-voting [4] utilizes blind signatures to verify whether ballots are submitted by legitimate voters without revealing its generator. All these traditional protocols require a trusted third party.

Zhao *et al.* [11] introduced blockchain into e-voting systems in 2015. Since then, more blockchain-based e-voting schemes were proposed, some of which still used a trusted third party to complete their scheme, such as [12]. Some of the existing work, e.g., [13], are based on cryptocurrencies such as Bitcoin [6]. McCorry *et al.* [14] presented the first implementation of a decentralized and self-tallying internet voting protocol in 2017, while the voting result cannot be revealed if one voter refuses to cast his/her ballot in the voting phase. These self-tallying e-voting systems need users to do the calculation job both on-chain and off-chain, making the system user-unfriendly.

III. System Model, Security Assumptions, and Design Goals

This section introduces the system model, security assumptions, design goals of the proposed AvecVoting.

A. System model

1) The structure of the voting system: As shown in Fig. 1, AvecVoting system is divided into three layers:

- **User layer:** There are two kinds of users: initiators and voters. Users log into the voting system after identity authentication and use the services provided by the application layer.
- **Application layer:** The application layer consists of applications that provide services to users, including Gen.Ballots to generate ballots for voters, Vote.Main to submit and count ballots, Committee.Select to select the counter committee, and Counter.Payoff to pay or punish counters.
- **Data layer:** The data layer is essentially a blockchain that supports smart contracts, such as Ethereum. All related data are maintained through smart contracts on the blockchain, so that authorized participants can reliably get their required information.

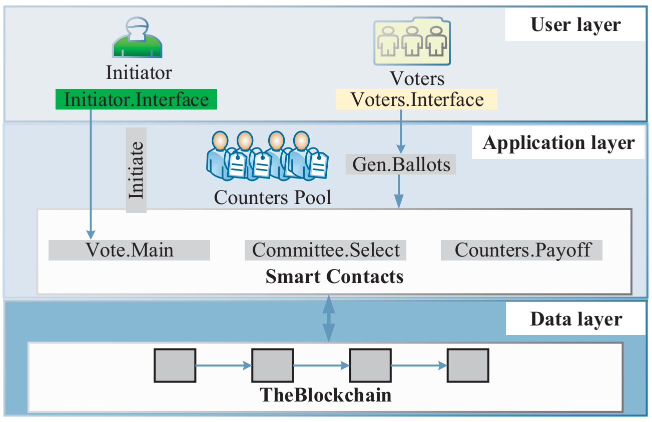


Fig. 1. The structure of the voting system.

2) Entity information:

- **Initiator:** The initiator is responsible for establishing smart contracts, publishing smart contracts and related parameters, and submitting voting fees. It is supposed to have a blockchain account.
- **Voter:** Voters are responsible for the key negotiation and voting. To establish the system public key and restore the private key of the threshold cryptosystem, voters should have secure channels between each other.
- **Counter:** Any user on the blockchain can become a counter by adding his/her blockchain wallet address $w.address$ into the $CountersPool$. During the counting stage, the smart contract invokes the random selection algorithm to select some counters from CountersPool to form a counter committee (CC), which is responsible for verifying the legality of ballots and counting all the legal ballots. Each counter in CC should have a certain wallet balance to use as deposits and his/her reputation value is greater than 0. These requirements will be described in detail in IV-C.

B. The security assumption

First of all, we assume that all entities are curious, i.e., they want to know the intermediate results of the voting process, including the content of each voter's ballot, the calculation or decryption results of a specific step, or whether a voter has submitted a valid ballot, *et al.* In addition, each entity may have the following malicious behaviors during its operations.

Authorized licensed use limited to: University of Science & Technology of China. Downloaded on August 27,2022 at 06:55:52 UTC from IEEE Xplore. Restrictions apply.

- **Initiator:** A malicious initiator may try to publish wrong smart contracts. However, since all the contents of a smart contract are publicly accessed, a wrong smart contract will be discovered soon, and the corresponding malicious initiator will also be exposed. Therefore, we do not consider the malicious initiator in AvecVoting.
- **Voter:** A malicious voter may try to submit illegal ballots, e.g., incorrectly formatted ballots, out-of-range ballots, or repeatedly ballots when one vote per person is restricted. Besides, according to AvecVoting's threshold encryption, we assume that a small number of voters are negative and do not provide their secret shares during the counting stage.
- **Counter:** We assume that counters are untrustworthy, means that they may provide wrong verifying or counting results. However, we also assume that most of them aim to profit by performing their tasks honestly, because providing wrong results will damage their interests.

C. Design goals

AvecVoting aims to realize the following security and performance design goals.

- **Anonymity and Confidentiality:** The identity of voters and the corresponding relationship between voters and ballots will not be disclosed. No one can obtain intermediate voting results before the end of the voting stage.
- **Correctness and Verifiability:** Correctness requires that all valid ballots are counted in the final result and all invalid ballots are discarded. So as to ensure that the final result of counting votes is the sum of legal ballots. Verifiability includes two aspect: each participant can verify whether a ballot is valid and whether the final voting result is correct. Specifically, a ballot will be identified as illegal if it is generated by a illegal voter or by a voter who have already submitted a ballot. Illegal ballots, incorrectly formatted ballots and out-of-range ballots are invalid. Otherwise, the ballot is valid.
- **Robustness and Practicability:** AvecVoting should have robustness and practicability. Robustness requires that even when some entities fail to complete their tasks, the system can still operate correctly. Practicability requires AvecVoting to complete voting and counting within acceptable overhead that meets the real-world demand.

IV. The Proposed AvecVoting

A. Overview

AvecVoting system is an anonymous and verifiable e-voting system based on blockchain. It adopts threshold encryption and one-time ring signature to satisfy basic security requirements of e-voting systems and adopts blockchain to realize reliable counting with untrustworthy counters. AvecVoting system consists of three main entities, i.e., initiators, voters, counters, and three stages, i.e., initialization, voting, and counting. After that, there is also a payoff process to pay or punish the counters.

Firstly, the initiator initiates a single-choice vote, sets options, and adds each legal voter's ring public key into the legal ring public key sets PK^R for subsequent ballot verification. Besides, legal voters conduct key negotiation off-chain to obtain the threshold public key pk^T and calculate the secret shares for subsequent decryption and counting. Next is the voting stage, where voters decide their choice, encrypt the ballots, generate one-time ring signatures, and submit ballots and signatures to the smart contract. After the voting stage, voters submit their secret shares to the smart contract, i.e., Vote.Main, for subsequent ballot decryption. Vote.Main invokes Committee.Select to run RandomSortition algorithm to randomly select a group of counters as CC, the counter Committee, which is responsible for verification and counting. Counters in CC first restore the threshold private key sk^T which can decrypt the ballots. They then verify the one-time ring signature of each ballot and use sk^T to decrypt it if legal. After decrypting all the legal ballots, counters then count the valid ballots and submit the final result to Vote.Main. When all counters in CC have submitted the result or timeout, Vote.Main then counts the received results and adopts the most numerous result as the final voting result. Finally, Vote.Main invokes PayOff contract to reward honest counters and punish malicious ones.

It should be noted that AvecVoting system can reduce the calculation burdens of initiators and voters by using counters to verify and count the ballots. Furthermore, the designed algorithms such as RandomSortition and PayOff ensure that AvecVoting can achieve correct counting even with some malicious counters.

B. The three stages of AvecVoting

This section will introduce the specific process of AvecVoting system, which is divided into three stages: 1) initialization and key negotiation, 2) voting, 3) verification and counting.

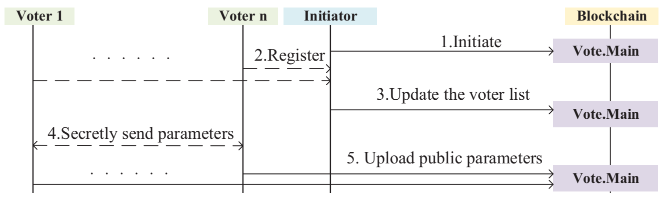


Fig. 2. Initialization and key negotiation.

1) Initialization and key negotiation: The initialization and key negotiation stage consists of the following five steps, as shown in Fig. 2, and solid arrows represent on-chain interactions, dashed arrows represent off-chain ones.

- *Initialization.* The initiator initializes Vote.Main contract by setting parameters including $Topic$, number of voting options N, deadline for voter registration t_{reg}, for secret share submitting t_{ss}, for voting t_v and ballot counting t_c, the number of voters n_v and counters n_c, the least number of secret shares to restore the threshold private key n_{ss}, revenue and fine for counters m_r, m_f, etc., and initiating the following lists L_{option}, L_{voter}, $L_{key-img}$, L_{ballot}, $L_{counter}$ and paying the cost of this voting m_v.

Authorized licensed use limited to: University of Science & Technology of China. Downloaded on August 27,2022 at 06:55:52 UTC from IEEE Xplore. Restrictions apply.

- *Voters registation.* For $i \in \{1, ..., n_v\}$, $Voter_i$ registers with the initiator and submits the identity ID_i and ring public key pk_i^R to the smart contract Vote.Main.
- *Updating the voter list.* Then $Vote.Main$ updates L_{voter}. After t_{reg}, the smart contract $Vote.Main$ will no longer accept registration information.
- *Secretly sending parameters.* $Voter_i$ randomly selects an element $x_i \in F_q$ and generates a polynomial $f_i(z)$ of degree $n_{ss}-1$ as $f_i(z) = \sum_{j=0}^{n_{ss}-1} f_{i,j} z^j \mod q$ where $f_{i,0}=x_i$. Then, $Voter_i$ calculates $s_{i,j}=f_i(j)$, for $j = \{1, ..., n_v\}, j \neq i$ and secretly sends $s_{i,j}$ to $Voter_j$ through an off-chain secure channel. After receiving other voter's parameters $s_{j,i}$, $Voter_i$ calculates his/her secret share by $s_i = \sum_{j=1}^{n_v} s_{j,i}$.
- *Uploading public parameters.* $Voter_i$ calculates $F_{i,0} = f_{i,0}G$ where G is the base point of the elliptic curve of threshold crypto system, and uploads it to $Vote.Main$.

2) Voting: For $i \in \{1, ..., n_v\}$, $Voter_i$ views L_{option} during the voting stage and generates a ballot B_i' according to his/her choice and calculates the threshold public key by $pk^T = \sum_{i=1}^{n_v} F_{i,0}$. Then $Voter_i$ uses $Enc(pk^T, B_i')$ to encrypt his/her ballot B_i' and get an encrypted ballot B_i, where pk^T is the threshold public key. Then $Voter_i$ uses his/her ring private key and others' ring public key to generate a ring signature S_i on B_i by $S_i = (I_i, c_i^1, ..., c_i^{n_v}, r_i^1, ..., r_i^{n_v})$, where I_i is $Voter_i$'s key image to prevent duplicate signatures, c_i^j and r_i^j are obtained with other voters' help to realize anonymity. Here the calculation process is omitted due to space limitations, and [15] can be referred for more details. Subsequently, $Voter_i$ submits $B_i||S_i$ to Vote.Main. Vote.Main checks whether the ring signature of the ballot is first used by checking whether the key image I is existed. If so, Vote.Main updates the ballot to L_{ballot}, or discards it if not. After t_v, $Vote.Main$ will no longer accept ballots.

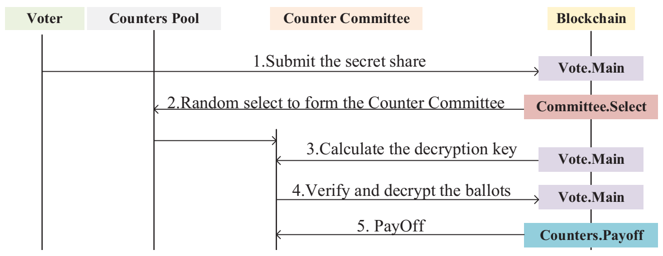

Voter
Counters Pool
Counter Committee
Blockchain
1.Submit the secret share
Vote.Main
2.Random select to form the Counter Committee
Committee.Select
3.Calculate the decryption key
Vote.Main
4.Verify and decrypt the ballots
Vote.Main
5. PayOff
Counters.Payoff

Fig. 3. Verification and counting.

3) Verification and counting: As shown in Fig. 3, the processes of verification and counting stage are as follows.

- *Submitting secret share.* When counting starts, each $Voter_i$ submits the secret share s_i to Vote.Main. After t_{ss}, $Vote.Main$ no longer accepts secret shares.
- *Randomly forming the counter committee.* Counters in the CountersPool who want to participant in the counting should adjust their status to online. Vote.Main invokes Committee.Select to randomly select a group of counters from online counters as CC, the counter committee.
- *Calculating the decryption key.* After CC is selected, each counter in it, denoted as $Counter_k$ $(1 \leq k \leq n_c)$, calculates the threshold private key sk^T by

$$sk^T = \sum_{i=1}^{n_{ss}} s_i \cdot \prod_{j=1, j\neq i}^{t} \frac{j}{j-i} \mod n_G,$$

where n_G is the order of G.
- *Verifying and decrypting ballots.* $Counter_k$ verifies each ballot B_i in L_{ballot} by $Ver(B_i, S_i)$. If legal, $Counter_k$ then decrypts each B_i by $Dec(sk^T, B_i)$ and counts all the decrypted valid results and generates the final counting vector $Result_k = (sum_1, ..., sum_N)$, where sum_n $(1 \leq n \leq N)$ represents the votes of the nth option. Then $Counter_k$ submits $Result_k$ to $Vote.Main$. After t_c, $Vote.Main$ no longer accepts counting results and it determines the most numerous counting vector $Result = (sum_1, ..., sum_N)$ and publish it as the final result RV.
- *PayOff.* For incentive, after obtaining RV, $Vote.Main$ invokes Counters.Payoff contract to reward the counters who submitted the correct result, that is $Result_k = RV$, and punish others who were wrong.

C. Details of Main Algorithms

Algorithm 1: Random Sortition

```
1  assert($oc >= 10 \cdot N$);
2  $seed \leftarrow hash_{block}$;
3  $SC \leftarrow \emptyset$;
4  $i \leftarrow 0$;
5  while $i < N$ do
6      $index \leftarrow seed \% len(RC)$;
7      if $RW[index].state == Online$ &&
8      $RW[index].reputation > 0$ &&
9      $RW[index].balance > fine$ then
10         $RW[index].state \leftarrow Candidate$;
11         $oc - -$;
12         Add $RC[index] \Rightarrow SC$;
13         i++;
14     end
15     $seed \leftarrow hash(seed)$;
16 end
17 return $SC$;
```

1) Random Sortition: The algorithm RandomSortition is used to select a group of independent and non-collusive counters from $CountersPool$ to form the counter committee CC. The input is a list of counter addresses RC with length $len(RC)$, number of online counters oc (oc is required to be ten times greater than $len(RC)$ to guarantee randomness), number of CC (N), and the hash value of the latest block's timestamp and difficulty $(hash_{block})$. The output is an address list that represents the members of CC. As shown in Algorithm 1, the algorithm RandomSortition takes $hash_{block}$ as the

Authorized licensed use limited to: University of Science & Technology of China. Downloaded on August 27,2022 at 06:55:52 UTC from IEEE Xplore. Restrictions apply.

seed (i.e., $seed$) to prevent cheating. A counter can be selected into CC only when its serial number is equal to $seed$ and has a positive $reputation$ value and balance more than m_f. The seed of each iteration is the hash of the previous one. This process repeats until n_c members are selected.

2) PayOff: PayOff is responsible for reward the honest counters and punish malicious ones to promote counters to complete the counting process correctly. As in Algorithm 2, counters who gave a correct result are rewarded a bonus m_r, and others lose their penalty deposit m_f as a fine. In addition, PayOff introduces the parameter $reputation$, which is recorded in the smart contract Counters.Payoff, to prevent malicious counters from being selected into CC. A counter's $reputation$ value increases when he/she counted correctly and decreases otherwise. If a counter's $reputation$ value becomes less than zero, he/she can no longer participate in the counting task. Through such an incentive mechanism, according to the Nash equilibrium, only the honest counter can get the maximum benefit [16]. Since we consider that most counters aim to benefit by participating in counting tasks, the final counting result is considered to be true.

Algorithm 2: Payoff

```
1  $i \leftarrow 0$;
2  while $i<len(CC)$ do
3      if $CC[i].result == RV$ then
4          $CC[i].balance \leftarrow CC[i].balance+m_r$;
5          $CC[i].reputation \leftarrow CC[i].reputation + 1$;
6      end
7      else
8          $CC[i].balance \leftarrow CC[i].balance-m_f$;
9          $CC[i].reputation \leftarrow CC[i].reputation - 2$;
10     end
11 end
```

V. Security and performance analyses

A. Security analysis

In this section, we analyze possible attacks and misbehaviors and explain how AvecVoting can resist these attacks.

1) Attacks against anonymity:

- **Adversaries trying to match voters with ballots:** $Voter_i$'s signature on his/her ballot B_i has the form of $S_i = (I_i, c_i^1, ..., c_i^n, r_i^1, ..., r_i^n)$, where I_i doesn't expose $Voter_i$'s public key, and c_i^j and r_i^j confuse $Voter_i$ with other voters in the voter group. Therefore, given S_i, one can only confirm whether $Voter_i$ is in the voter group, but cannot determine his/her specific identity [10].
- **Voters trying to obtain information before submitting ballots:** Each ballot is encrypted with the threshold public key pk^T, and only when counting starts and more than n_{ss} voters submit their secret shares can the private key sk^T be restored [9]. Since counting starts after the finish of the voting stage, it is impossible for any voter to obtain any information before submitting the ballot.

2) Attacks against Correctness and Verifiability:

- **Voters repeatedly voting:** The key image I_i is used to prevent repeated voting. To generate a valid S_i, $Voter_i$ must calculate I_i as required, and I_i in $Voter_i$'s different signatures are the same. When $Vote.Main$ receives two ballots with a duplicate I_i, it will discard the latter one.
- **Adversaries pretending to be a legitimate voter:** During the registration stage, the initiator adds all the valid voters' ring public key pk_i^R $(1 \leq i \leq n_v)$ to $Vote.Main$. Later when voting, voters are supposed to sign the ballot with their ring private key sk_i^T. However, an adversary who have no ring private key cannot generate a valid signature, thus the ballot that he/she submits will be discarded by counters during counting stage.
- **Adversaries trying to tamper with ballots:** Due to the non-tamperable nature of blockchain, no one can tamper with the ballot once it is submitted to $Vote.Main$.
- **Some counters submit wrong counting results:** The counter committee CC is selected by RandomSortition which uses the timestamp and difficulty value of the current block as the $seed$. Due to the unpredictability of the next block, counters have no way to conclude. Besides, the PayOff incentive mechanism promotes counters to count correctly and the reputation limitation of CC also prevent malicious counters to be selected. Thus, AvecVoting can achieve correct counting even when some counters are untrustworthy.

3) Attacks against robustness:

- **Some voters not providing their secret shares:** Among the n valid voters, as long as n_v voters submit their secret shares, CC can then recover the threshold private key sk^T and decrypt ballots. Since n and n_v are set by the vote's initiator during the initialization stage, we consider that the initiator is able to select appropriate values to ensure the successful decryption.

B. Performance analysis

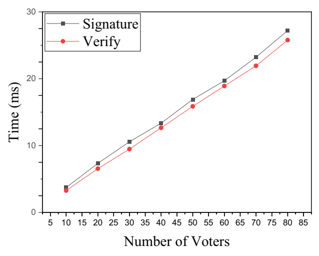
1) The off-chain part: To guarantee performance, AvecVoting completes the encryption & decryption and signature & verification processes off-chain, and we therefore conduct the experiments of this part off-chain. Our experiments are conducted on a virtual machine running Ubuntu 20.04 with Intel Core i5 8500 CPU, 4GB RAM. In our experiments, we set the number of voters varies from 10 to 80. As shown in Fig. V-B1, the cost of signature & verification increases approximately linearly with the increase in the number of voters. The cost of encryption & decryption increases as the number of voters increases. Furthermore, the proportion of the decryption cost increases as the value of t increases. This is because that the computing power required to recover the threshold private key sk^T increases as the threshold value n_{ss} increases.

2) The On-chain part: For the on-chain part, we conduct our experiments with Ethereum smart contracts written in Solidity and deployed on an Ethereum test network. The

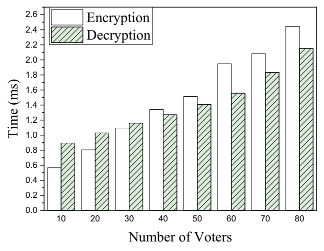
Authorized licensed use limited to: University of Science & Technology of China. Downloaded on August 27,2022 at 06:55:52 UTC from IEEE Xplore. Restrictions apply.

TABLE I
ON-CHAIN CONSUMPTION OF EACH ROLE

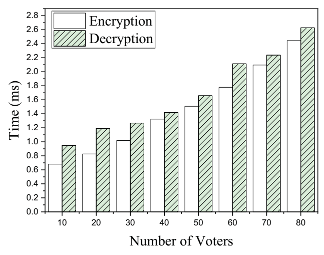
Role	Theoretical Analysis of the Transaction Cost	Gas Cost	ETH Cost
Initiator	$C_{\text{Initialization}}$	453939	0.0091
Voter	$C_{\text{Regist}} + C_{\text{GetVoterConfig}} + (n-1) \cdot C_{\text{GetFi}} + C_{\text{UploadFi}} + C_{\text{SubmitBallot}} + C_{\text{UploadShare}}$	1623587	0.0325
Counter	$C_{\text{GetVoterConfig}} + C_{\text{RandomSelect}} + n \cdot (C_{\text{GetShare}} + C_{\text{GetBallot}}) + C_{\text{UploadResult}} + C_{\text{PayOff}}$	2987579	0.0597



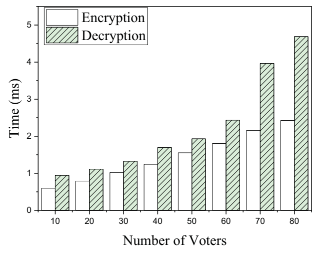
(a) The cost of signature & verification.



(b) The cost of Enc & Dec, t=0.5n.



(c) The cost of Enc & Dec, t=0.6n.



(d) The cost of Enc & Dec, t=0.7n.

Fig. 4. The cost off-chain

theoretical analyses of the transaction cost and gas cost and Ether cost of each role are listed in TABLE I where C_{Event} represents the on-chain cost of to execute Event. The gas and ETH cost shows that counters undertake more computing work, which is reasonable because counters have the stronger computing power and can get rewards through counting. Assigning computing work to counters dramatically reduces the computational overhead of voters and initiators, making AvecVoting user-friendly and more practical. We can also see that the initiator has the least on-chain cost, which is reasonable because the initiator should pay the counters.

VI. Conclusion

In this paper, we proposed AvecVoting, an blockchain-based e-voting system that provides both strong security and high performance. For basic security requirements of e-voting, we utilized threshold encryption and one-time ring signature to protect voters' privacy and ballots' confidentiality. For improving performance and practicability, we designed the counter model for efficient counting utilizing smart contracts. Specifically, we proposed the RandomSortion algorithm and the reputation-based PayOff incentive mechanism to ensure that AvecVoting can achieve correct counting even when some counters are untrustworthy. Finally, we gave security and performance analyses, which showed that AvecVoting provides strong security while solving the performance issues caused by blockchain, providing good efficiency in both voting and counting stages.

Acknowledgment

This work is supported in part by the National Natural Science Foundation of China under Grant No. 61972371, and Youth Innovation Promotion Association of the Chinese Academy of Sciences (CAS) under Grant No. Y202093.

References

[1] S. E. Adekunle *et al.*, "A review of electronic voting systems: Strategy for a novel," *International Journal of Information Engineering & Electronic Business*, vol. 12, no. 1, pp. 19–29, 2020.
[2] C. A. Neff, "A verifiable secret shuffle and its application to e-voting," in *Proceedings of the 8th ACM Conference on Computer and Communications Security (CCS)*. ACM, 2001, pp. 116–125.
[3] X. Fan, T. Wu, Q. Zheng *et al.*, "HSE-Voting: A secure high-efficiency electronic voting scheme based on homomorphic signcryption," *Future Generation Computer Systems*, vol. 111, pp. 754–762, 2020.
[4] M. Kumar, S. Chand, and C. P. Katti, "A secure end-to-end verifiable internet-voting system using identity-based blind signature," *IEEE Systems Journal*, vol. 14, no. 2, pp. 2032–2041, 2020.
[5] N. Szabo, "Smart contracts," 1994, accessed on 2022-02-16. [Online]. Available: https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html
[6] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," https://bitcoin.org/bitcoin.pdf, 2008, accessed on 2022-02-16.
[7] S. Park, M. Specter, N. Narula, and R. L. Rivest, "Going from bad to worse: from internet voting to blockchain voting," *Journal of Cybersecurity*, vol. 7, no. 1, pp. 1–15, 2021.
[8] Y. G. Desmedt, "Threshold cryptography," *European Transactions on Telecommunications*, vol. 5, no. 4, pp. 449–458, 1994.
[9] T. P. Pedersen, "A threshold cryptosystem without a trusted party," in *Proceedings of the 1991 Workshop on the Theory and Application of of Cryptographic Techniques (Advances in Cryptology-EUROCRYPT)*. Springer, 1991, pp. 522–526.
[10] P. Mundhe, V. K. Yadav, A. Singh *et al.*, "Ring signature-based conditional privacy-preserving authentication in VANETs," *Wireless Personal Communications*, vol. 114, no. 1, pp. 853–881, 2020.
[11] Z. Zhao and T.-H. H. Chan, "How to vote privately using bitcoin," in *Proceedings of the 2015 International Conference on Information and Communications Security (ICISC)*. Springer, 2015, pp. 82–96.
[12] W. Zhang, Y. Yuan, Y. Hu *et al.*, "A privacy-preserving voting protocol on blockchain," in *Proceedings of the 11th IEEE International Conference on Cloud Computing (CLOUD)*. IEEE, 2018, pp. 401–408.
[13] S. Bistarelli, M. Mantilacci, P. Santancini, and F. Santini, "An end-to-end voting-system based on bitcoin," in *Proceedings of 32nd Annual ACM Symposium on Applied Computing (SAC)*. ACM, 2017, pp. 1836–1841.
[14] P. McCorry, S. F. Shahandashti, and F. Hao, "A smart contract for boardroom voting with maximum voter privacy," in *International Conference on Financial Cryptography and Data Security*. Springer, 2017, pp. 357–375.
[15] N. van Saberhagen, "Cryptonote v 2.0," 2013, Menero white paper, Accessed on 2022-02-16. [Online]. Available: https://bytecoin.org/old/whitepaper.pdf
[16] H. Zhou, X. Ouyang, Z. Ren *et al.*, "A blockchain based witness model for trustworthy cloud service level agreement enforcement," in *Proceedings of the 2019 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2019, pp. 1567–1575.

Authorized licensed use limited to: University of Science & Technology of China. Downloaded on August 27,2022 at 06:55:52 UTC from IEEE Xplore. Restrictions apply.